RAPID DESIGN & CONTROL OF SMALL UNMANNED AERIAL SYSTEMS

A Thesis

by

JUSTIN DAVID BARNES

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirement for the degree of

MASTER OF SCIENCE

Chair of Committee,	Raktim Bhattacharya
Committee Members,	Moble Benidict
	Sivakumar Rathinam
	John Valasek
Head of Department,	Rodney D.W. Bowersox

December 2016

Major Subject: Aerospace Engineering

ABSTRACT

As the Federal Aviation Administration (FAA) begins to allow commercial Unmanned Aircraft (UA) flights in the United States, businesses and organizations of all types are exploring a wide variety of potential applications. When compared to manned alternatives, these UA have the capability to reduce cost, improve efficiency, expand capabilities, and increase safety. One class of UA that shows particular promise is small Unmanned Aerial Systems (sUAS). These aircraft have lower takeoff weights and smaller form factors than manned alternatives. Yet they often have similar (or better) data-gathering capabilities at a fraction of the price. The benefits of this technology and the corresponding demand is clear. However, while unmanned aircraft are relatively low cost, they can still be a large expense. This is due to the high development, production, and implementation costs that frequently accompany flight vehicles.

One possibly significant source of expense in autonomous vehicle design is the development of the control system. This work presents a compilation of design methods, when taken as a whole, serve to reduce the time and expense of designing a control system for sUAS. The modeling methods presented include vehicle dynamics, vehicle aerodynamics, propeller aerodynamics, and electric motor dynamics. The modern H_∞ control design method was used with the resulting high-fidelity 6 DOF model to produce controllers for hover and forward flight configurations of a tiltrotor sUAS.

I dedicate this work to my wife, my parents, and grandmother. They have supported me through these years with great love and care.

I am also thankful to the Lord of lights for providing more than I could ever need. As I finish this work, I am mindful of James 1:17 which reads: ”Every good gift and every perfect gift is from above, coming down from the Father of lights, with whom there is no variation or shadow due to change.” Also, from Acts 17:28 ”In him we live and move and have our being...”

ACKNOWLEDGMENTS

I would like to thank Dr. Raktim Bhattcharya and Dr. Mobel Benedict for their advisement during this work.

I would also like to acknowledge the support given by MathWorks and their comprehensive software tools. Matlab, Simulink, and SimMechanics (now Simulink Multibody) were instrumental in achieving the design goals of this work.

NOMENCLATURE

BDC	Brushed Direct Current
BLDC	Brushless Direct Current
CAD	Computer Aided Design
CFD	Computational Fluid Dynamics
DOF	Degree of Freedom
ESC	Electric Speed Controller
FAA	Federal Aviation Administration
FEA	Finite Element Analysis
MIMO	Multiple Input Multiple Output
NAS	National Airspace System
PSP	Pilot Support Package
RHP	Right Half Plane
SISO	Single Input Single Output
sUAS	small Unmanned Aerial System
UA	Unmanned Aircraft
VLM	Vortex Lattice Method

TABLE OF CONTENTS

LIST OF FIGURES

1 INTRODUCTION

As was mentioned in the Abstract, the main contribution of this research is the compilation of rapid design and modeling methods which facilitate modern control synthesis for sUAS. The methods were chosen to be rapidly implementable, provide sufficient accuracy, and reduce cost.

One possibly significant source of expense is development and tuning of the vehicles control system. While classical methods are applied in industry on all sorts of systems (including sUAS [1]), a considerable limitation is that they can only be applied to single-input-single-output (SISO) systems. Since an aircraft system has multiple inputs and outputs (MIMO), classical approaches require the designer to divide and decouple the MIMO system into a number of SISO systems. This decoupling can lead to inaccurate modeling since the interactions of the coupled plant states can be lost. Therefore, this method of modeling and control design can lead to a mismatch between the real-world system and the implemented controller. This then can cause the need for in-field tuning of the controller to achieve acceptable performance. Some argue that the ability to tune classic controllers in the field is beneficial because it provides flexibility. The argument can also be made that if the plant model is accurate enough and the design method robust enough, in-field tuning will not be necessary.

Another significant source of expense while modeling aircraft performance is wind tunnel testing. This time consuming and expensive process results in accurate static aero-

dynamic behavior but will be shown to be unnecessary for H_∞ control design.

For these reasons and more, there is growing interest in applying optimal control [2] and specifically H_∞ Control [3] to small unmanned aircraft. This work is an addition to this effort in that it is a compilation of modeling and design methods that facilitate the synthesis of modern controllers for sUAS. It is important to note the generality of the following methods in that they can be applied to a wide range of vehicle types and configurations. Whether the vehicle be a conventional airplane, flying wing, blended body, pusher, tractor, or tiltrotor vehicle, the process is the same

The methods mentioned above will be described in detail as they are applied to an example tiltrotor sUAS in the following sections. The entire vehicle system was divided into four distinct subsystems: vehicle mass properties, vehicle aerodynamics, propeller aerodynamics, and electric motor dynamics. These subsystems were then compiled in one environment in order to create a comprehensive, nonlinear, 6 DOF model. The MathWorks toolboxes Simulink and SimMechanics 1st Generation were chosen as that single design environment. Together, Simulink and SimMechanics provide a block diagram programming language in which mechanical systems can be modeled and simulated. sections 2 through 5 will describe how each subsystem was created and implemented within the vehicle system. Section 6 will then describe the control design for both cruise and hover configurations of the vehicle.

2 VEHICLE MASS PROPERTIES MODEL

Three dimensional computer aided design (CAD) software was utilized during the design process to build an accurate representation of the mass proprieties of the vehicle as is shown below in Figure 2.1.

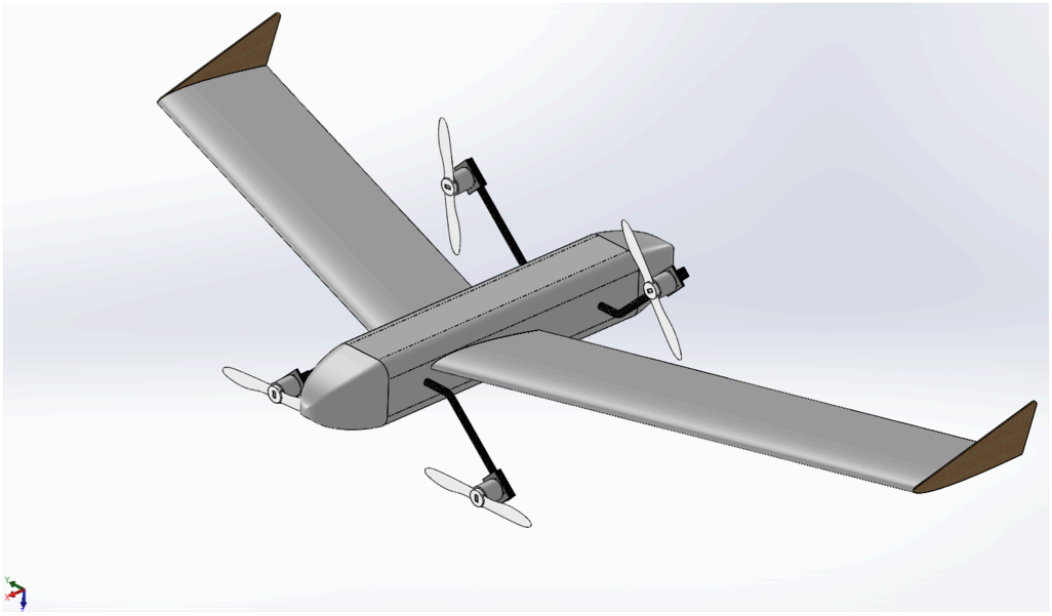

Figure 2.1: CAD Model of Tiltrotor sUAS

This vehicle is an example of a tiltrotor sUAS capable of VTOL and transition to efficient fixed wing forward flight. For a vehicle of this type, the gyroscopic and inertial effects the four propulsion units have on the aircraft dynamics are complex. Of course, this derivation can be done by hand but would be time consuming; especially if it had to be remade every time the vehicle configuration was altered. Therefore, alternatives were explored in an effort to preserve method generality and reduce design time. A solution was found in SimMechanics. This is a MATLAB toolbox which interfaces with Simulink and

allows for mechanical systems of rigid bodies to be simulated with relative ease. Adding to the functionality is another toolbox called SimScape Multibody Link which allows 3D CAD files (from supported 3D modeling software) to be imported into SimMechanics. In this process, each separate part in the CAD model is converted to a SimMechanics Body Block. Joint relations between these rigid bodies can also be imported if they are constrained correctly in the CAD model.

An important note in the import process is that there will be as many Body Blocks in the imported model as there are parts in CAD model. Since there are many small components (sensors, processors, wiring, batteries, etc.) included in the CAD model, these will all be represented as separate body blocks which tend to complicate the SimMechanics model. Yet, there is no need to preserve these component's individuality when they will not have relative motion with other body blocks. For this reason, all of the components which are stationary with respect to the fuselage of the aircraft were grouped into one main body with a single set of mass properties. Those components that will be actuated and have relative motion were left as separate body blocks. This was accomplished by measuring the mass, location of Center of Gravity, and Inertia Tensor for the original model shown in Figure 2.1 and then assigning these properties to the main body of a simplified CAD model shown below in Figure 2.2.

It may seem comical that the aerodynamic shape of the aircraft has been reduced to a rectangular block with flat faces and sharp corners. Yet, this shape is only an easily modeled visualization. The effect the mass properties have on the vehicle dynamics is

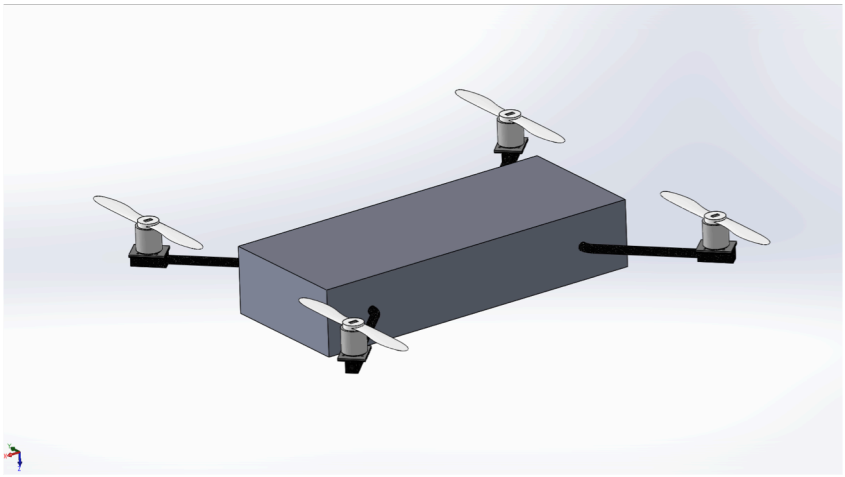

Figure 2.2: Simplified sUAS CAD Model

the only concern here. The simplified model with appropriate mass proprieties is then imported via SimScape Multibody Link. The resulting SimMechanics visualization and block diagram are shown in figures 2.3 and 2.4.

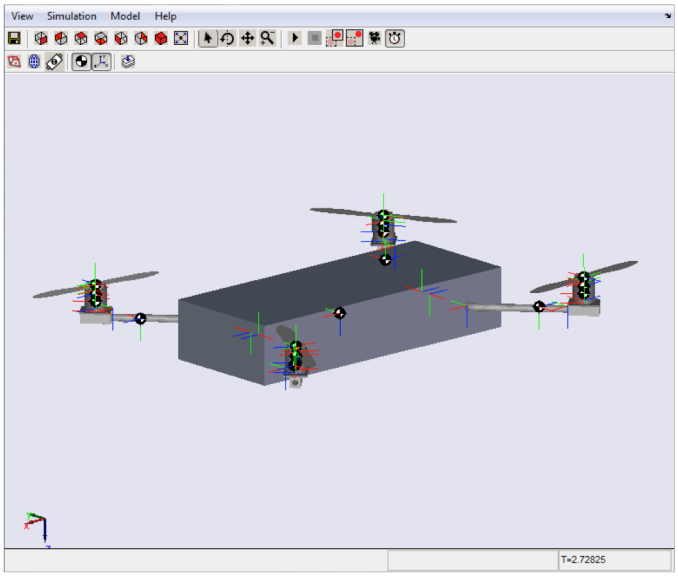

Figure 2.3: Simplified sUAS Model in SimMechanics Visualization Window

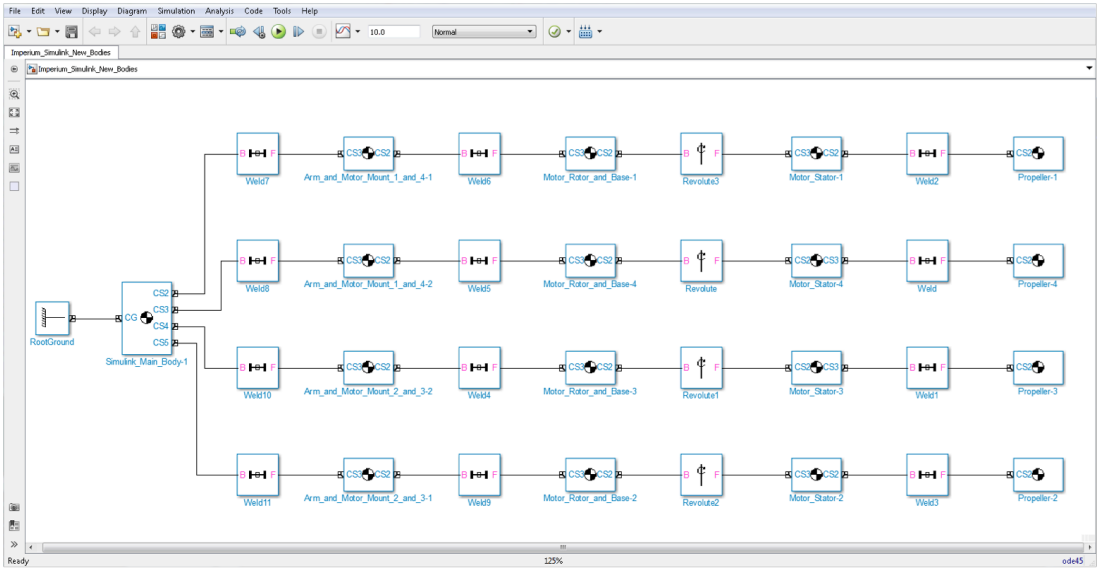

Figure 2.4: SimMechanics Block Diagram after SimScape Multibody Link Import

As is shown in Figure 2.4, the main body block is connected to a ”RootGround” block on the left and to blocks on the right which make up the arms, motors, and propellers. Either a Weld or Revolute joint relation block can be seen between each body block. These joint blocks can be replaced with a wide variety of joint relations included in SimMechanics. These include universal, planer, and 6 DOF joints. In addition each body and joint block can be actuated with both internal and external forces and moments. In this way, SimMechanics provides the framework to apply the propulsive and aerodynamic forces which will act on the aircraft and simulate the resulting dynamics. If changes are made to the physical design, these can be updated easily within the SimMechanics blocks or by re-importing the CAD model. There is no need to re-derive equations of motion.

2.1 Note on Structural Design

The structural components of the flight vehicle were chosen with ease of manufacturing, budget, and structural resilience in mind. While advanced Finite Element Analysis (FEA) could be used to find a more optimal structure, the complexity of this analysis was avoided for multiple reasons. First, this is another attempt to reduce cost by minimizing the number of expensive software packages required. Secondly, the emphasis placed on weight savings when designing large aircraft is often not necessary for sUAS. This is largely due to the range that is required and the difference in the rate of fuel consumption for the two classes of aircraft. Consider the example of a large, long range aircraft such as a Boeing jet. If the aircraft weight can be reduced by 1%, this will translate to considerable fuel savings over one year of flight. Yet if the structure of a sUAS is over-designed such that the weight increases by 1%, the effect will likely never be noticed by the operator. Finally, the ability of a sUAS to withstand a hard landing or impact without major structural damage is extremely desirable. Depending on the mission requirements, this structural resilience or "toughness" can be more important than shaving weight for range or endurance. That being said, composites are to be used throughout the design because of their high strength to weight characteristics.

3 VEHICLE AERODYNAMICS MODEL

A key goal of this research is to avoid the need for the most accurate (and expensive) aerodynamic testing and simulation. For this reason, computationally expensive computational fluid dynamics (CFD) methods which solve versions of the Navier-Stokes Equations were avoided in preference to the inviscid approach called Vortex Lattice Method (VLM). This method is attractive because it can be implemented in software to be very computationally light while still producing relatively accurate results. The performance of this method will be discussed further after a brief description of the foundational principles of VLM.

3.1 Description of Vortex Lattice Method

The major assumption of Vortex Lattice Method is that it deals with Linear Aerodynamics. Linear Aerodynamics occur at low Mach Numbers and low angles of attack - where compressibility and stall effects can be neglected. Even with this limitation, this method can still prove very useful since aircraft spend considerable time in the linear region of flight. Also, even though VLM is less than perfect, it will be shown by wind tunnel test that this method can be impressively close.

The mathematical formulation of VLM is built around the idea of vortex lines. These lines, which are shown above in Figure 3.1 [4], approach the trailing edge of the lifting surface from infinity, turn sharply to pass along the quarter-chord line of a so called panel, and

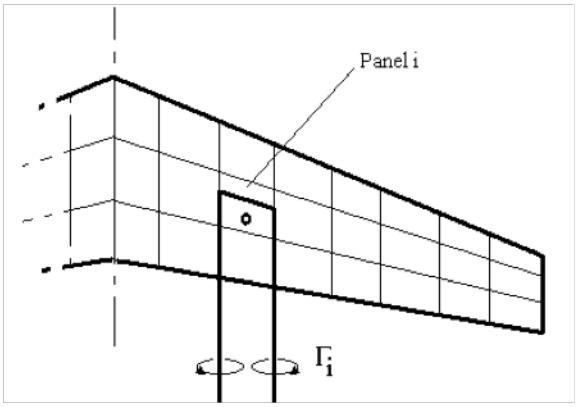

Figure 3.1: Vortex Line and Collocation Point for Panel “i”

then extend back out to infinity. There is also a so called collocation point, represented as a dot on ”Panel i” in Figure 3.1, where boundary conditions for the panel will be enforced. The lifting surface is divided into many such panels and each panel has its own vortex line and collocation point defined in this manner. A flow field is induced about the lifting surface by the vorticity associated with each vortex line and is represented by the vortex strength Γ_i. This induced velocity can be used to calculate the force on each panel using the Kutta-Jukovski Theorem [5] as is shown in Equation 3.1.

$$\overline{F} = \rho * (\overline{V}_{ind} \times \overline{\Gamma}) \cdot l \tag{3.1}$$

Where ρ is the air density, $\overline{V}_{ind}$ is the induced velocity, $\overline{\Gamma}$ is the vortex strength, and l is the length of the vortex segment along the quarter-chord line. This theory can be implemented in the software language of choice.

3.2 Athena Vortex Lattice Method

The implementation of VLM which was chosen for this research is Athena Vortex Lattice Method (AVL) [6]. This program allows aircraft configurations made of multiple lifting surfaces to be created (using text files) and then simulated with an executable file called from Command Prompt (on Windows machines). The results can also be written to text files for post-analysis and documentation.

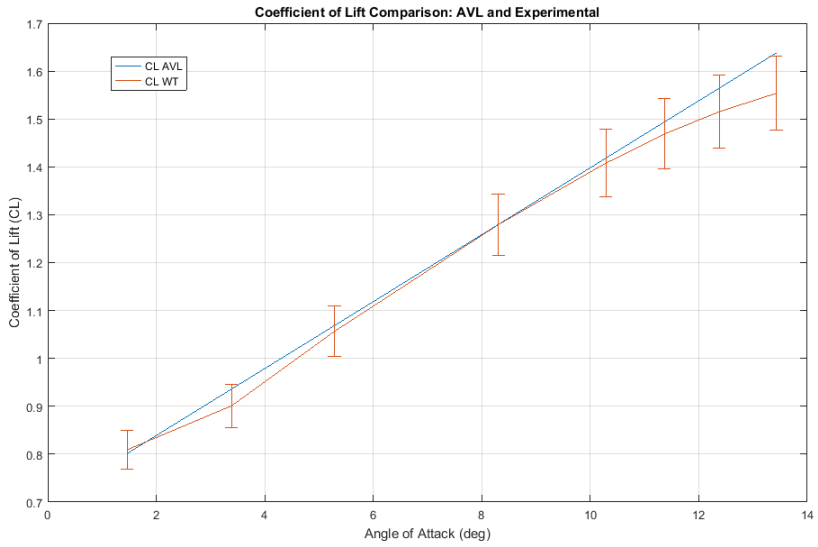

Figure 3.2: Comparison of AVL Prediction and Wind Tunnel Test Results

As was mentioned, VLM is capable of sufficient accuracy in the linear regions of flight. Figure 3.2 shows the coefficient of lift versus angle of attack of a highly cambered wing designed by the author. Wind tunnel test data is shown in orange with 5% error bars and the AVL prediction is shown in blue. The AVL prediction stays withing 5% error until approximately 13 degrees angle of attack when the airfoil is nearing stall. These results give confidence in AVL's ability to predict lift and lift-induced drag in the linear region of

flight.

3.3 Implementation of AVL in Simulink Model

The aerodynamics model of the vehicle was incorporated into Simulink by creating look-up tables for that the simulation could reference at each time-step. These tables allow for much faster simulation than if the aerodynamic coefficients were calculated during simulation. This proved to be a wise choice while debugging and when simulating the effect of a controller on the non-linear model.

In total, six look up tables were generated: three force and three moment. These tables take seven flight parameters as inputs and output force and moment coefficients calculated at the CG of the aircraft about each body axis. While it is convention to express aerodynamic forces and moments in the Wind Axis coordinate system, when implementing the actuation blocks in Simulink, it was found to be more convenient if the forces and moments were measured in the fixed body frame as is shown in Figure 3.3. The *World* coordinate frame is stationary with respect to the ground. The *Body* coordinate system is fixed to the vehicle in the conventional manner with the x axis out the nose, y axis out the starboard (right) wing, and z out the bottom of the aircraft.

Conveniently, AVL provides the choice to calculate forces and moments in both Body or Wind axis, saving the need for a coordinate transformation in post processing. In fact, only one inconvenience was found while using AVL: it does not offer the batch-run capability that was necessary for creating the large aerodynamic look-up tables. The solution

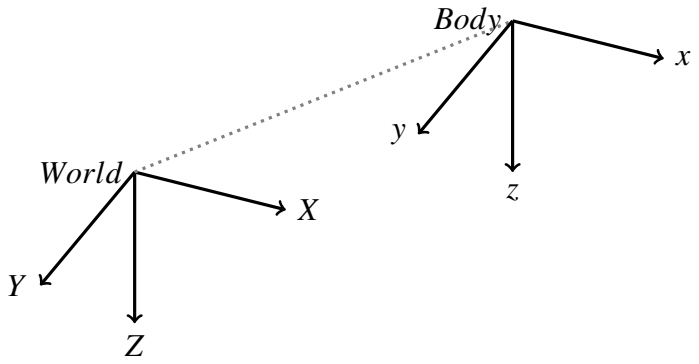

Figure 3.3: World and Body Fixed Coordinate Systems

was to write a custom script. AVL operates off of a lifting surface geometry configuration file (.avl) and two optional files which specify the mass properties (.mass) and run cases (.run). The .run file is usually written by the AVL executable file and serves as a set of instructions when running more than one case. This allowed the author to create a program that created a lengthy version of this file that would serve in place of a batch-run capability. The last challenge was post processing the data. After running each case, the output is stored in a text file. Another program was written to read the desired results from these text files and create the appropriate look-up tables.

Each look-up table is implemented in the Simulink environment using the n-D Lookup block and is basically a 7-dimensional grid of nodes. Each node represents the force or moment coefficient at a flight condition which is specified by the seven aerodynamic parameters measured at that simulation step. As is shown in Figure 3.4, these seven inputs are the angle of attack of the wing (alpha), the sideslip angle (beta), three non-dimensional angular rates of the vehicle (pb/2v, qb/2v, and rb/2v), elevator deflection , and aileron deflection. If the specified flight condition does not lie exactly on a node,

the value of the coefficient is interpolated. It is extremely convenient that AVL uses the velocity magnitude to non-dimensionalize the angular rates of the vehicle. This not only allows the look-up tables to have one less input, but this also reduces the time it takes to compute the tables since an entire table dimension is absorbed into the angular rates.

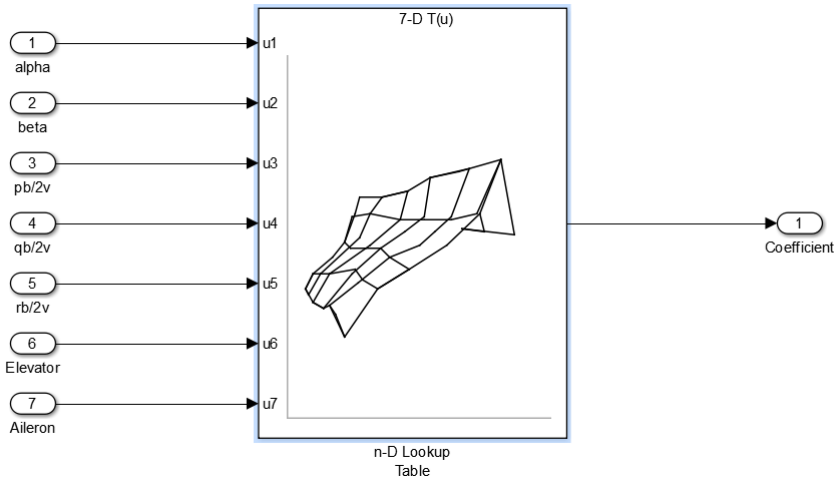

Figure 3.4: n-D Lookup Table Example

The resulting coefficients are then used to calculate the forces and moments for each body axis using Equations 3.2 through 3.7.

$$F_X = q_\infty S C_{F_X} \tag{3.2}$$

$$F_Y = q_\infty S C_{F_Y} \tag{3.3}$$

$$F_Z = q_\infty S C_{F_Z} \tag{3.4}$$

$$M_X = q_\infty S b C_{M_X} \tag{3.5}$$

$$M_Y = q_\infty S c C_{M_Y} \tag{3.6}$$

$$M_Z = q_\infty S b C_{M_Z} \tag{3.7}$$

Where S is the reference wing area, b is the reference wingspan, c is the reference chord length, and the dynamic pressure is defined:

$$q_\infty = \frac{1}{2}\rho V_\infty^2 \tag{3.8}$$

Where ρ is the air density and V_∞ is the free-stream velocity. These aerodynamic forces and moments were then applied to the CG of the main body shown in Figure 2.2 using Body Actuator Blocks.

4 PROPELLER AERODYNAMICS MODEL

4.1 Method Description

Modeling the propeller aerodynamics was approached by applying the modern Lift and Drag Equations as are defined in [7] and expressed:

$$L = q_\infty S C_L \tag{4.1}$$

$$D = q_\infty S C_D \tag{4.2}$$

Where C_L and C_D are the coefficients of lift and drag respectively, S is the reference area, and the dynamic pressure q_∞ is defined as in Equation 3.8. These equations can be written in span-wise sectional form as:

$$l = q_\infty c C_l \tag{4.3}$$

$$d = q_\infty c C_d \tag{4.4}$$

Where c is the chord length, and C_l and C_d are the sectional (two-dimensional) lift and drag coefficients respectively. These equations can be thought of as the lift and drag per span. Equations 4.3 and 4.4 can be applied to a wing in rotary motion, such as a propeller blade, by recognizing that velocity, chord length, and angle of attack will all vary along the span of the blade. Now referencing the span-wise direction as the radius r, we will formulate the lift and drag contribution of each infinitely thin cross-section dr along the blade from R_{hub} to R_{tip}, as is shown in 4.1.

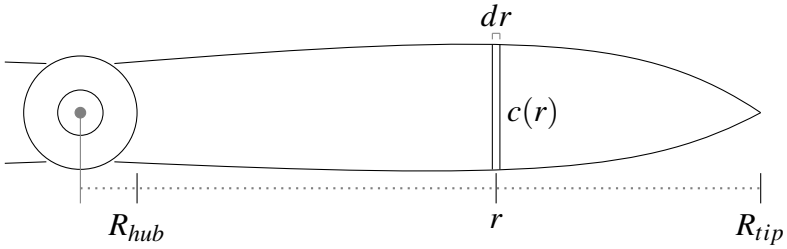

Figure 4.1: Propeller Blade Dimensions

Now examining the airfoil cross-section located at r:

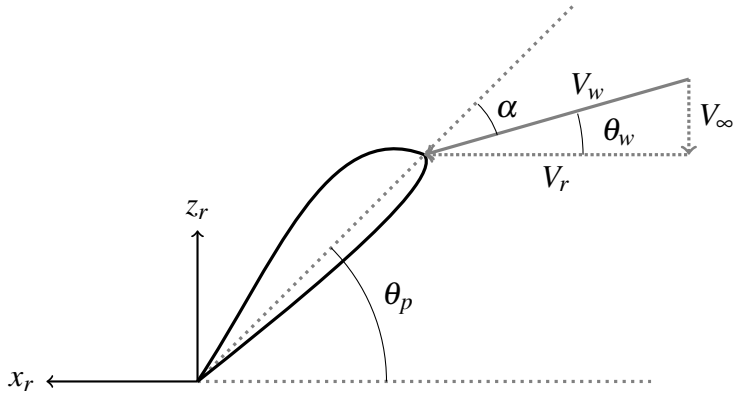

Figure 4.2: Propeller Airfoil Cross Section

We define the free-stream velocity which flows normal to the arc of the propeller V_∞. The velocity induced upon the airfoil by the rotation of the propeller is V_r. These two component velocities combine to produce the wind velocity V_w. This wind velocity defines the conventional wind coordinate system and its angle above the propeller arc will be called θ_w. The angle of attack α is defined to be the angle between wind direction and the chord line of the airfoil. Lastly the pitch angle θ_p is measured between the chord line of the airfoil and the propeller arc direction. Thrust will be in the z_r direction and the damping force on the propeller will be in x_r direction.

Using these definitions the lift and drag equations can be developed as follows. The wind velocity can be expressed as:

$$V_w = \sqrt{V_\infty^2 + V_r(r)^2} \tag{4.5}$$

The angle of attack can be expressed:

$$\alpha = \theta_p(r) - \theta_w(r) \tag{4.6}$$

The angle of the wind axis can be expressed in terms of the free-stream velocity and rotational velocity:

$$\theta_w = \tan^{-1}(\frac{V_\infty}{V_r(r)}) \tag{4.7}$$

And writing the rotational velocity V_r in terms of the angular velocity of the propeller ω and radius r we have:

$$V_r = r\omega \tag{4.8}$$

Now θ_w is known in terms of r. Shifting attention to θ_p, it is assumed in this work that the propeller is an air-screw, meaning there is a combination of free-stream velocity V_∞ and rotational speed ω which will cause the propeller to be at zero angle of attack at all locations along the radius. The function that satisfies this relationship is:

$$\theta_p = \tan^{-1}(\frac{\Delta x}{2\pi r}) \tag{4.9}$$

Where Δx is the distance the propeller travels through the air in one revolution with no slip (commonly called the pitch of the propeller). θ_p is also commonly called the Lead Angle in mechanical design and applies to all types of helical shapes such as screws and worm

gears. Substituting Equations 4.6, 4.7, and 4.8 into 4.5 we have:

$$\alpha = \tan^{-1}(\frac{\Delta x}{2\pi r}) - \tan^{-1}(\frac{V_\infty}{r\omega}) \tag{4.10}$$

Using the following trigonometric identity:

$$\tan^{-1}(x) \pm \tan^{-1}(y) = \tan^{-1}(\frac{x \pm y}{1 \mp xy}) \tag{4.11}$$

And simplifying, we finally have an expression for the angle of attack in terms of r:

$$\alpha = \tan^{-1}(\frac{\Delta x\omega - 2\pi V_\infty}{2\pi r\omega + \frac{\Delta x V_\infty}{r}}) \tag{4.12}$$

Now we need to express the coefficients of lift and drag in terms of r. We can do this by finding a relationship between the coefficients and the angle of attack. This is common practice in aerodynamics and the shape of functions $C_l = f(\alpha)$ and $C_d = f(\alpha)$ are well known. These functions can be determined very accurately for a given airfoil shape. For this work, the following piecewise function for the lift curve of the selected airfoil was created.

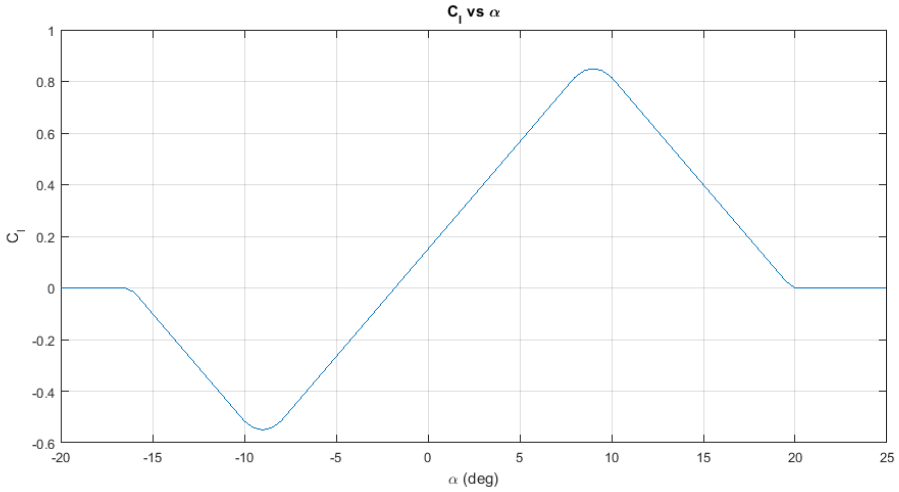

Figure 4.3: Airfoil Lift Curve

As is shown in 4.3, the airfoil has a maximum lift coefficient of approximately 0.85. It is important to note that for very high or low angles of attack, the lift coefficient is set to zero. This is done in case the simulation inputs a large positive or negative value, the propeller model will not become unstable because of unrealistically large lift values. This practice also seems reasonable since at high enough angles of attack (i.e. near 90 degrees), the wing will act like an air-break in that it will produce negligible lift and a large amount of drag.

Another important point is that, by using sectional lift curves in this manner, stall effects are captured and accurately represented at each radial location along the blade. Therefore, the model can predict the realistic situation when the propeller blade is stalled in some locations but not in others.

The implemented sectional drag curve is displayed in Figure 4.4. The drag bucket is centered at 1.5 degrees angle of attack with minimum coefficient of 0.0127.

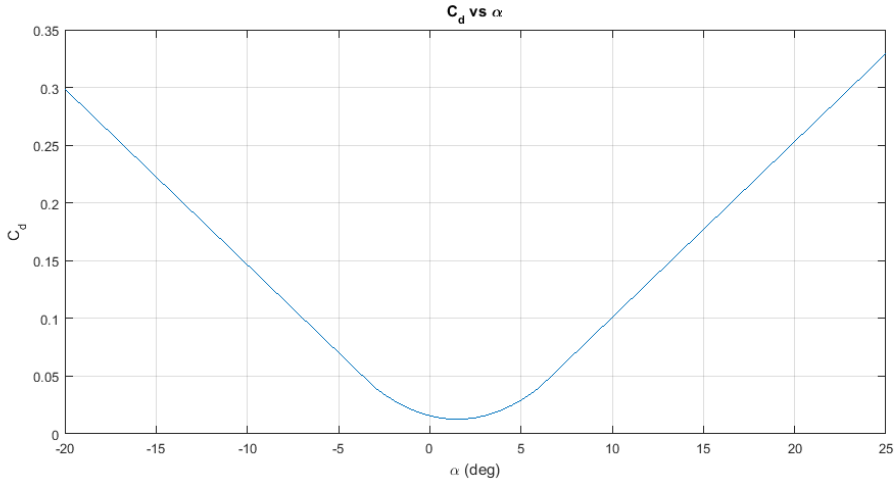

Figure 4.4: Airfoil Drag Curve

Rewriting the dynamic pressure in terms of Equation 4.5:

$$q_\infty = \frac{1}{2}\rho[V_\infty^2 + V_r(r)^2] \tag{4.13}$$

Now the sectional lift and drag equations (4.3 and 4.4) can are rewritten in terms of r:

$$l = q_\infty(r)c(r)C_l(\alpha(r)) \tag{4.14}$$

$$d = q_\infty(r)c(r)C_d(\alpha(r)) \tag{4.15}$$

Where $c(r)$ was implemented by taking measurements at multiple points along the span of the actual propeller and then using a curve fit function to estimate the chord length at each radial location. Converting sectional lift and drag forces into the x_r and z_r directions (Figure 4.2), thrust on one propeller can now be calculated by integrating along r:

$$F_z = \int_{R_{hub}}^{R_{tip}} [l(r)cos(\theta_w(r)) - d(r)sin(\theta_w(r))]dr \tag{4.16}$$

Sectional lift and drag can also be multiplied by r and then integrated to yield the torque

due to drag about the hub due to one propeller:

$$M = \int_{R_{hub}}^{R_{tip}} [l(r)sin(\theta_w(r)) + d(r)cos(\theta_w(r))]rdr \tag{4.17}$$

Now let n be the number of blades on the selected propeller. The total thrust and torque due to drag is:

$$T = nF_z \tag{4.18}$$

$$Q = nM \tag{4.19}$$

Finally, in a similar manner to the vehicle aerodynamics, look-up tables were created for a specific propeller using a custom MATLAB function. The look-up tables take the free-stream velocity V_∞ and the angular rotation rate of the propeller ω as inputs (since all other parameters are geometric constants for a given propeller). The outputs are of course the thrust and torque due to drag.

4.2 Results and Discussion

The resulting thrust produced by the model is shown in Figures 4.5 and 4.6 below.

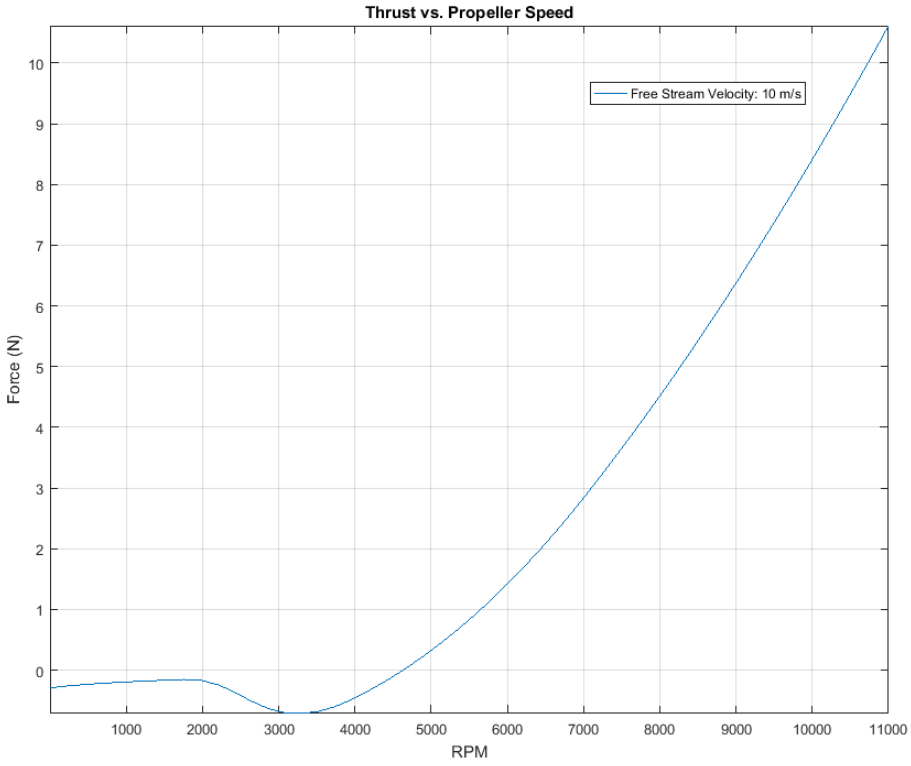

Figure 4.5: Thrust vs. Propeller RPM (Fixed Free Stream Velocity of 10 m/s)

Figure 4.5 shows the thrust verses the propeller RPM. Small negative thrust is produced at zero RPM since only the angle of attack is low and only the sectional drag is contributing (Equation 4.16). This is observed until approximately 2000 RPM when parts of the airfoil begin to have a negative lift coefficient (the angle of attack reaches approximately -16 degrees, see Figure 4.3). At this fixed free stream speed, the thrust increases until the max RPM of approximately 11,000.

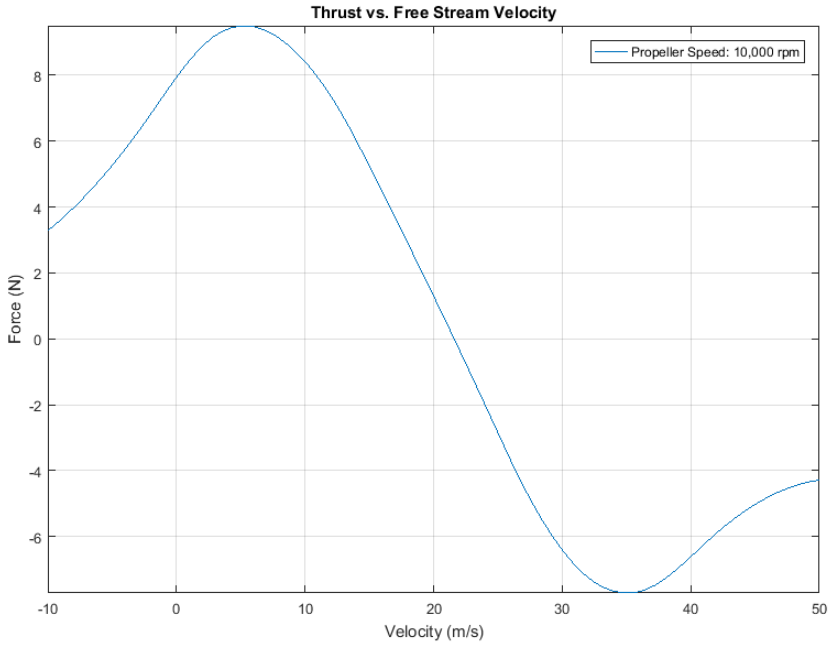

Figure 4.6: Thrust vs. Free-Stream Velocity (Fixed Propeller Speed of 10,000 rpm)

Figure 4.6 shows the thrust verses free-stream velocity while the propeller is kept at a constant angular speed of 10,000 rpm. The results show that the propeller produces a maximum thrust of approximately 10 Newtons (2.25 lbs) when traveling at a speed of 5 m/s and ceases to provide thrust at a speed of approximately 23 m/s. As the velocity continues to increase, the angle of attack will become negative. Eventually (after 50 m/s) the model will produce close to zero thrust again as the coefficient of lift becomes zero and the coefficient of drag is the only thrust contribution below -16 degrees angle of attack.

The major assumptions and limitations of this model are as follows:

1. The density of the air is constant and the flow entering the propeller is laminar.
2. There is no radial (or span-wise) flow along the propeller blades.

3. Induced velocities caused by blade-tip vortices are not considered.

The assumption of laminar flow is perhaps most questionable during hover close to obstacles such as walls or the ground. The obstacles would of course increase the chance of recirculation of the flow.

However, assuming laminar flow when the vehicle is in cruise configuration is reasonable. Careful design consideration was made to ensure that the wing downwash effect and propeller wash from the forward propellers have a minimal effect on the aft propellers.

Ignoring induced velocities will result in this model over-predicting thrust and under-predicting drag since the real system loses energy to vortices and recirculating flow.

5 ELECTRIC MOTOR MODEL

5.1 Method Introduction

The motors chosen during the vehicle design process were Brushless DC (BLDC) Motors. Compared to Brushed DC (BDC) Motors, BLDC motors are much more efficient at converting electric energy to mechanical energy. This provides a clear advantage when considering power to weight ratios of aircraft propulsion units and the endurance afforded. The disadvantage of this choice, in terms of this work, came when attempting to develop a model of the motor dynamics. While applying Newton's 2nd Law to the motor kinematics (no external forces or torques) is relatively simple, the control input voltage and it's effects are much more complex to model. This is because, unlike BDC motors, BLDC motors require a small computer called an Electric Speed Controller (ESC) to regulate the energy it receives from the battery. Modeling this interaction and control is considerably more complex than modeling a BDC motor. Therefore, in keeping with the purpose of minimizing engineering effort in the modeling process, a compromise was made.

In short, it was determined that from a control design perspective, it is important for the actuator model to have similar response to a given input in magnitude and frequency. Therefore, if the actuator model performance matches the real system, it does not matter that the internal working of the model match the real system. With this in mind, a BDC motor model was created and then parameter estimation techniques were used to match the performance of the modeled BDC motor to the real BLDC motor.

The reader may wonder at this point, if the internal system does not matter (as long as the correct performance was achieved), why another system of differential equations or transfer functions was not used instead of the BDC motor model. The reasoning behind this choice is that if a BDC motor model was made to match the performance of the real BLDC motor, insight would be gained about what motor performance parameters are necessary to gain the desired control. This will be discussed more in section 5.3.

5.2 Method Description

The method for creating a Simulink model for a BDC motor was found in reference [8]. Following this method, the mathematical model is created by summing the internal and external torques being applied to the motor. Applying Newton's Second Law we have:

$$J\ddot{\theta} + b\dot{\theta} = K_T i \tag{5.1}$$

Where $\dot{\theta}$ is the angular velocity of the motor, J is the moment of inertia, b is the damping coefficient due to friction, K_T is the torque constant specific to the motor, and i is the current passing through the motor.

Now applying Kirchhoff's Voltage Law we have:

$$L\frac{di}{dt} + Ri = V - K_e\dot{\theta} \tag{5.2}$$

Where L is the inductance of the motor, R is the resistance, V is the voltage input, and K_e is the Back EMF constant. These two differential equations can be modeled in a Simulink block diagram as is shown in Figure 5.1. The input is voltage supplied to the motor and the output is motor speed in rad/s.

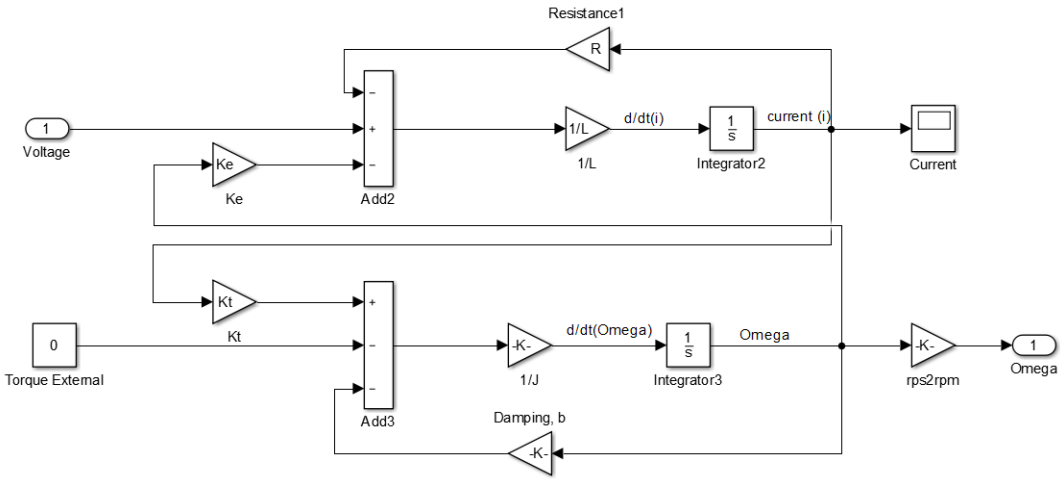

Figure 5.1: BDC Motor Simulink Block Diagram

Next, experimental data was gathered for the true BLDC motor to be used in the design. The full experimental setup will not be discussed here but a step input was provided to the motor and response was recorded as will be shown later in Figure 5.2.

The next task was to find the combination of the six coefficients in Equations 5.1 and 5.2 which cause the Simulink model shown in Figure 5.1 to match the response measured in the experiment. This was accomplished by using Parameter Estimation in Simulink as is described in reference [9]

5.3 Results and Discussion

The simulation results from the Simulink model (post parameter estimation) are compared to experimental data in Figure 5.2.

The simulated response of the BDC motor is a close, but not exact, match to the measured experimental data taken from the BLDC motor. While there is little difference in the steady state speed, the rise time is slightly slower. This was judged to be acceptable

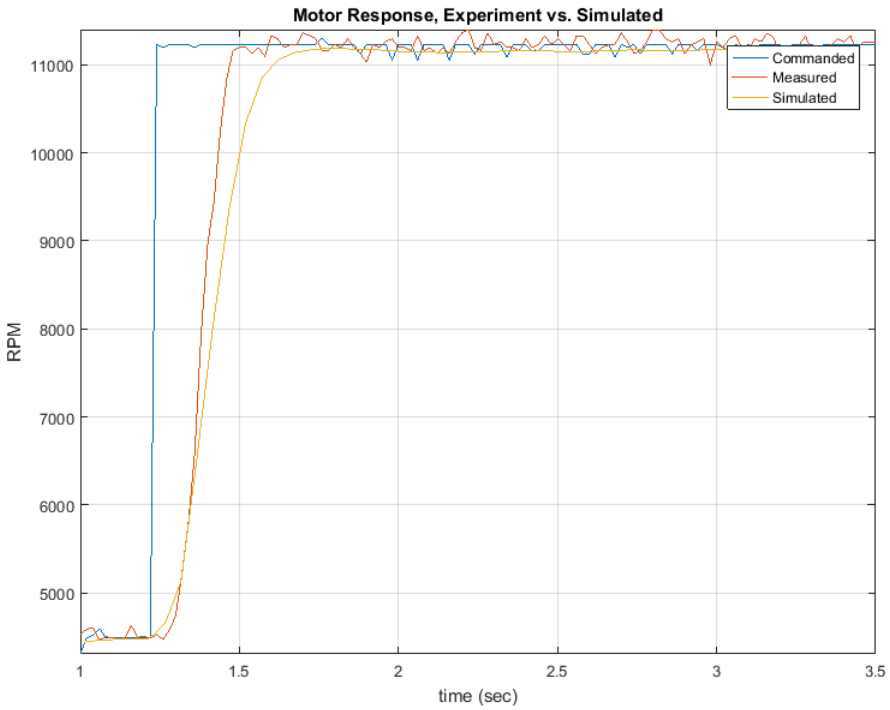

Figure 5.2: DC Motor Step Response, Measured vs. Simulated

since the modeled performance is more than sufficient to render adequate control of the system.

It is interesting to note that, for the BDC model to match BLDC performance, the BDC model had to pull considerably more current. This effect can be observed in the real world by comparing BDC and BLDC performance and is therefore a reasonable result.

6 MODERN CONTROL DESIGN IMPLEMENTATION

6.1 Control Method Selection

Today there are numerous control design methods from which a control engineer can choose. Classical methods are a commonly applied in industry since the mathematics involved is (usually) relatively simple and the methods are widely applicable. However, while the formulation of classical methods makes them well suited for single-input-single-output (SISO) systems, their application to systems with multiple inputs and outputs (MIMO) is less straightforward. Since an aircraft system is certainly MIMO, classical designers often divide the MIMO system into a number of SISO systems. However, if not performed carefully, this method can lead to modeling inaccuracies since the interactions (coupling) between the plant states can be lost. Therefore, this method can lead to a mismatch between the real-world system and the implemented controller. Causing the need for excessive in-field tuning of the controller to achieve acceptable performance.

Alternatively, modern H_∞ control design has the following key advantages:

1. It is formulated for the MIMO case with no decoupling necessary.
2. It incorporates a sense of optimality in terms the system norm.
3. Not only can error be minimized, but also system states and control effort.
4. It does not require full-state feedback, which adds to it's utility when controlling a variety of systems.
5. Rapid synthesis with no need for tuning.

For these reasons, H_∞ control design was the method of choice for this work. However, it is by no means the only modern or optimal method which could be applied. Given the complexity of tiltrotor aircraft, extensive work could be performed on the control system alone. However, the main objective of this research was to demonstrate a control design method which is both applicable to the modeling methods described in the previous sections and is also consistent with rapid control design.

6.2 H_∞ Control Discussion

The first step in the control synthesis is to find a trim point in the desired flight regime (i.e. hover, level cruise, banking turn, etc.) and extract a conventional linear state-space model of the form:

$$\dot{x} = Ax + Bu \tag{6.1}$$

$$y_m = Cx + Du \tag{6.2}$$

Where x is the state vector, u is the control signal vector, and y_m is the measurement vector. Next, the conventional state-space model is reformulated with the addition of the vector z which is minimized during the H_∞ synthesis. Rewriting in matrix form we have:

$$\begin{bmatrix} \dot{x} \\ z \\ y_m \end{bmatrix} = \begin{bmatrix} A & B_1 & B_2 \\ C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{bmatrix} \begin{bmatrix} x \\ w \\ u \end{bmatrix} \tag{6.3}$$

Where w represents the exogenous inputs, and again, z represents the quantities to be minimized (often error e and control effort u). The standard block diagram for the optimal

control problem is shown below in Figure 6.1.

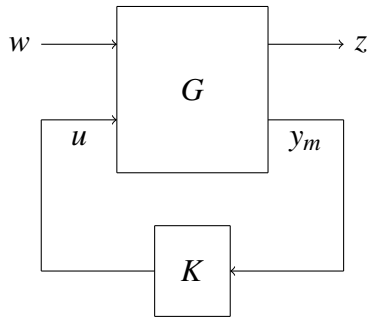

Figure 6.1: Standard Optimal Control Block Diagram

During the control synthesis, the H_∞ norm of the closed loop transfer function ($T_{w\to z}$) is minimized. In order to make sense of the size of this norm, the input and output signals must be normalized. The exogenous input w is normally made up of sensor noise n and disturbance d. Hence, the normalized noise ($\overline{n}$) and disturbance ($\overline{d}$) signals are created and can be represented as White Noise of unit intensity:

$$|\overline{n}| \leq 1 \tag{6.4}$$

$$|\overline{d}| \leq 1 \tag{6.5}$$

Weighted filters are then applied to these signals such that a realistic noise and disturbance signal is recovered and fed to the plant G:

$$|W_n\overline{n}| = n \tag{6.6}$$

$$|W_d\overline{d}| = d \tag{6.7}$$

In a similar way the output signals in z will likely need scaling in order to have:

$$|\overline{z}| \leq 1 \tag{6.8}$$

Therefore a filter must also be applied that satisfies:

$$|W_z z| = \bar{z} \tag{6.9}$$

Bear in mind these weighted filters could be low-pass, high-pass, band-pass, or notch to ensure the correct signal size is produced for the frequencies that will be encountered on the real system.

By use of these filters the entire closed loop system is normalized. This means that if the control synthesis can minimize the Infinity norm of the closed loop transfer function to be less than one:

$$\|T_{w\to z}\|_\infty \leq 1 \tag{6.10}$$

then the closed loop gain of the system is attenuated at all frequencies for all input-output pairs. If the system norm is greater than one, performance is being demanded which cannot be achieved. The capabilities of the system and the weighted filters must be realistic for the synthesis to be well-posed. For more information on the formulation of H_∞ control, please see reference [10].

6.3 The Linearized Model

For this work, two controllers were synthesized in total. One regulator for the vehicle in its hover configuration, and another for the vehicle's forward flight cruise configuration. Designing a control architecture for all phases of flight is beyond the scope of this work.

In order to obtain linear State-Space models for control synthesis, the full Simulink model was trimmed at the desired operation points (hover and cruise) and then linearized

using the MATLAB function $linmod()$. This function takes advantage of Jacobian derivatives which are preprogrammed into many Simulink blocks[11] and makes linearization a simple process. This functionality is one of many time saving advantages of representing a complex non-linear dynamic system in Simulink when performing control design.

The resulting state-space model contained 26 states which are listed below:

STATE	NAME	VARIABLE NAME
1-3	Vehicle 3D Position	P_x, P_y, P_z
4-6	Vehicle 3D Velocity	V_x, V_y, V_z
7-10	Vehicle Angular Position Quaternion	q_1,q_2, q_3, q_4
11-14	Vehicle Angular Velocity Quaternion	$\dot{q_1}$,$\dot{q_2}$,$\dot{q_3}$, $\dot{q_4}$
15-18	Propeller Blade Positions	P_{p1}, P_{p2}, P_{p3}, P_{p4}
19-22	Propeller Angular Velocity	ω_{p1}, ω_{p2}, ω_{p3}, ω_{p4}
23-26	Motor Currents	i_1, i_2, i_3, i_4

Table 6.1: sUAS System States

6.4 Control Synthesis: Cruise Configuration

The goal of the controller during forward flight was to regulate the Euler Angles (roll, pitch, and yaw) in the presence of wind-gust disturbance. Once the full state-space model was obtained, a minimum realization was sought for the regulator synthesis. The exogenous input was made up of the noise signal n and the disturbance signal d. Rewriting

equation 6.3 we have:

$$\begin{bmatrix} \dot{x} \\ z \\ y_m \end{bmatrix} = \left[\begin{array}{c|ccc} A & B_d & B_n & B_u \\ \hline C_z & D_{zd} & D_{zn} & D_{zu} \\ C_y & D_{yd} & D_{yn} & D_{yu} \end{array}\right] \begin{bmatrix} x \\ d \\ n \\ u \end{bmatrix} \tag{6.11}$$

For the cruise synthesis, the disturbance d is a 3-dimensional signal of wind-gusts in each of the body axis. The control input u is a 2-dimensional signal comprised of the Elevator and Aileron inputs. The Euler angles and the input voltages for the four motors were selected to be minimized in the vector z. Since these two sets of signals have differing magnitudes, they require separate weights. For this reason the z vector will be split into the state minimization vector z_s and the control signal minimization vector z_u. These signal choices and those for the measurement vector y_m are listed in Table 6.2. The reader may note that the angular position of the vehicle is expressed as quaternion states in Table 6.1 but in Euler Angles in Table 6.2. The reason being Simulink expresses angular states taken from a 6 DOF joint as quaternions. These were converted to Euler Angles in the measurement vector to make simulation results more readable and to facilitate the controller regulating these states to zero. If they were left in quaternion form, level flight would correspond to $q = [1,0,0,0]^T$. Regulating the output to a non-zero state such as the number 1 would require a reference input signal.

VECTOR	STATE NAME	VARIABLE NAME
z_s	Euler Angles	Φ, Θ, Ψ
z_u	Elevator and Aileron Inputs	δ_e, δ_a
y_m	Euler Angles	Φ, Θ, Ψ
	Vehicle Angular Rates	p, q, r
	Vehicle Velocity	V_x, V_y, V_z

Table 6.2: Control Synthesis Signals: Cruise

Rewriting Equation 6.11:

$$\begin{bmatrix} \dot{x} \\ z_s \\ z_u \\ y_m \end{bmatrix} = \left[\begin{array}{c|ccc} A & B_d & B_n & B_u \\ \hline C_{zs} & D_{11} & D_{12} & D_{13} \\ C_{zu} & D_{21} & D_{22} & D_{23} \\ C_y & D_{31} & D_{32} & D_{33} \end{array}\right] \begin{bmatrix} x \\ d \\ n \\ u \end{bmatrix} \tag{6.12}$$

Now the following simplifications can be made to the state-space model:

1. The noise signals only effect the measurement vector, therefore B_n, D_{12}, and D_{22} are all zero matrices of appropriate dimension.
2. The disturbance signal will only effect the states of the system, meaning D_{11}, D_{21}, and D_{31} are also zero matrices of appropriate dimension.
3. The control signal u will not effect z_s or y_m making D_{13} and D_{33} zero matrices as well.
4. C_{zu} is a zero matrix since D_{23} captures the effect of control input signal.

Substituting into Equation 6.12:

$$\begin{bmatrix} \dot{x} \\ z_s \\ z_u \\ y_m \end{bmatrix} = \left[\begin{array}{c|ccc} A & B_d & 0 & B_u \\ \hline C_{zs} & 0 & 0 & 0 \\ 0 & 0 & 0 & D_{23} \\ C_y & 0 & D_{32} & 0 \end{array}\right] \begin{bmatrix} x \\ d \\ n \\ u \end{bmatrix} \tag{6.13}$$

Now taking into account the true inputs and outputs of the system are the weighted signals expressed in equations 6.6, 6.7, and 6.9 we have:

$$\begin{bmatrix} \dot{x} \\ \overline{z_s} \\ \overline{z_u} \\ y_m \end{bmatrix} = \left[\begin{array}{c|ccc} A & W_d B_d & 0 & B_u \\ \hline W_{zs} C_{zs} & 0 & 0 & 0 \\ 0 & 0 & 0 & W_{zu} \\ C_y & 0 & W_{yn} & 0 \end{array}\right] \begin{bmatrix} x \\ \overline{d} \\ \overline{n} \\ u \end{bmatrix} \tag{6.14}$$

The reader should note that the weights placed on the inputs and outputs are scalar values, instead of frequency dependent filters as was mentioned previously. In this case, filtering was not necessary to achieve the desired performance in the controller.

Finally the MATLAB function $hinfsyn()$ was used to synthesize an H_∞ controller for the state-space model in Equation 6.14. The resulting closed-loop system was stable with an H_∞ norm of 0.0685. A simulation of the linear system with White noise injected in the disturbance channels is shown in Figures 6.2, 6.3.

The same controller is then applied to the full nonlinear model but instead of White Noise, the Dryden Gust Model is applied in each body axis with gusts of 2 m/s [12]. The results are shown in Figures 6.4 and 6.5. As expected, the linear system shows better performance, yet the nonlinear model is kept within 5 degrees rotation in all three axes. The

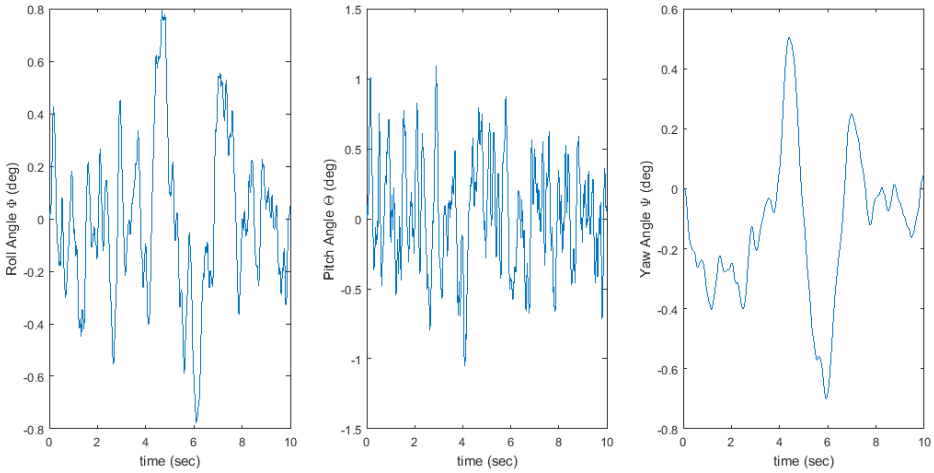

Figure 6.2: Linear Cruise Simulation Results: Euler Angles

necessary control input is very reasonable at only 2 degrees deflection from the Elevator channel and 1.5 degrees in the Aileron channel. This control synthesis shows that there is ample control authority to keep the vehicle in a desired angular position with only the control surfaces. Of course the four motors can provide torques in all three axes but they are not necessary for control unless the vehicle is in the hover configuration. The hover controller will be described next.

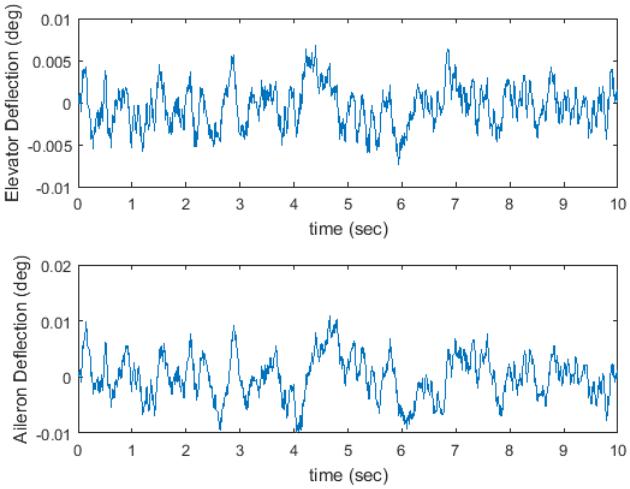

Figure 6.3: Linear Cruise Simulation: Control Surface Deflections

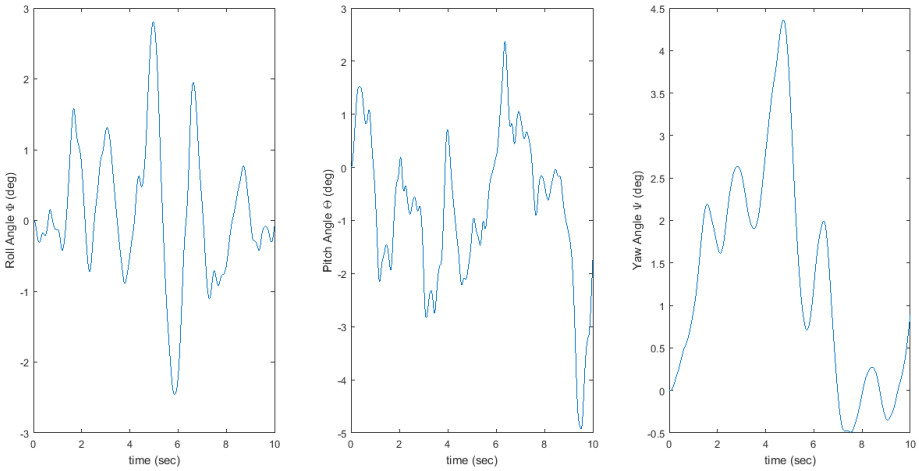

Figure 6.4: Nonlinear Cruise Simulation Results: Euler Angles

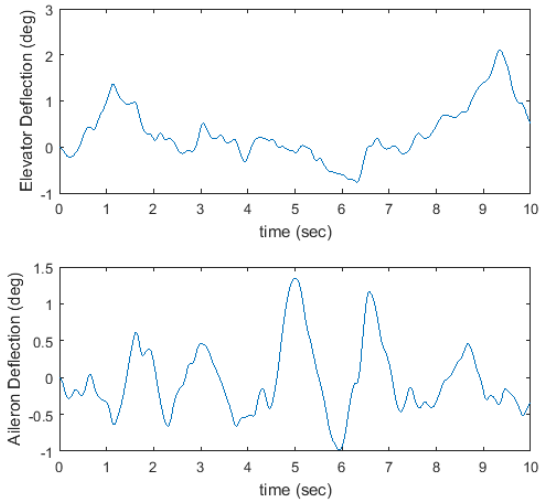

Figure 6.5: Nonlinear Cruise Simulation: Control Surface Deflections

6.5 Hover Controller Method Description

The hover controller for this vehicle was considerably more challenging to design than the cruise configuration. This is due to the presence of transfer function poles at the origin. The presence of these poles is caused by the physics of the hover configuration. Consider the system comprised of a mass tethered by one revolute joint located at its center of gravity. This case can be described with the following differential equation:

$$J\ddot{\theta} + b\dot{\theta} + k\theta = u \tag{6.15}$$

Where θ is the angular displacement, J is the moment of inertia about the axis of rotation, b is the damping coefficient, k is the spring coefficient, and u is the external torques applied to the system. Now consider that the damping coefficient and spring constant are negligible:

$$J\ddot{\theta} = u \tag{6.16}$$

Converting the system to the Laplace domain and solving for the transfer function we have:

$$\frac{\theta}{u} = \frac{1}{Js^2} \tag{6.17}$$

The resulting characteristic equation will have two poles at the origin of the $s - plane$.

It turns out that a vehicle in hover has similar behavior but in all three dimensions. Friction is negligible when rotating at a low speed in the air so there is no appreciable damping and there is certainly know spring behavior since the vehicle is not tethered. While the numerics which result from linearizing the Simulink model do not place these poles exactly at zero, the resulting six poles are very close ($1x10^{-10}$) to zero. This turns out to be a significant problem since modern synthesis techniques such as H_2 and H_∞ fail with

poles on the imaginary axis [10].

In attempt to fix this problem, the method described in reference [13] was implemented. This method removes the poles from the origin via a bilinear transformation:

$$s = \frac{\tilde{s} + p_1}{(\tilde{s}/p_2) + 1} \tag{6.18}$$

Graphically:

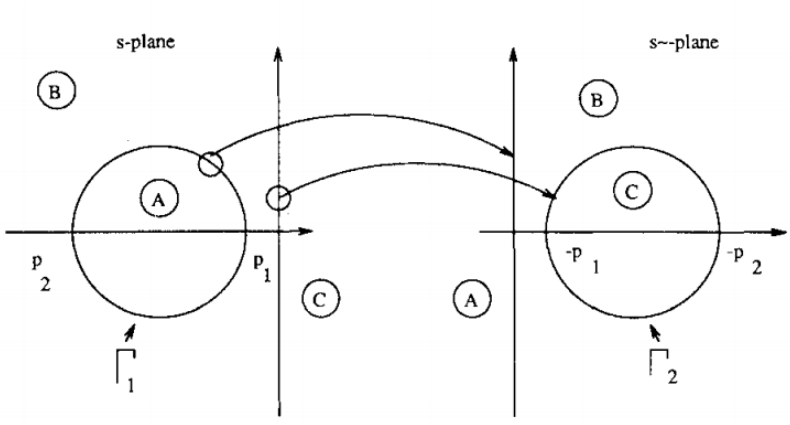

Figure 6.6: Bilinear Transform on s-plane

As is shown in Figure 6.6 [13], the poles that were located on the imaginary axis are now placed on the circle Γ_2 on the right half plane (RHP) of the $\tilde{s} - plane$. Now the control synthesis can occur in the $\tilde{s}$ domain with no poles on the imaginary axis, but this will produce a controller in $\tilde{s}$ domain as well. Therefore the inverse transform has to be applied to the resulting controller in order to apply it to the original problem.

$$\tilde{s} = \frac{-s + p_1}{(s/p_2) - 1} \tag{6.19}$$

6.6 Hover Controller Synthesis Results

The structure of the state-space representation for the hover configuration can be expressed by 6.14 as well. The difference in the two controllers is that now the motors are used for actuation since the control surfaces will not be effective with zero airspeed. Table 6.3 now reflects the new z_u.

VECTOR	STATE NAME	VARIABLE NAME
z_s	Euler Angles	Φ, Θ, Ψ
z_u	Voltage Inputs	V_1, V_2, V_3, V_4
y_m	Euler Angles	Φ, Θ, Ψ
	Vehicle Angular Rates	p, q, r
	Vehicle Velocity	V_x, V_y, V_z

Table 6.3: Control Synthesis Signals: Hover

The resulting controller was applied to the nonlinear system. For relatively short simulation times (up to 10 seconds), the results are acceptable, as is shown in Figures 6.7 and 6.8.

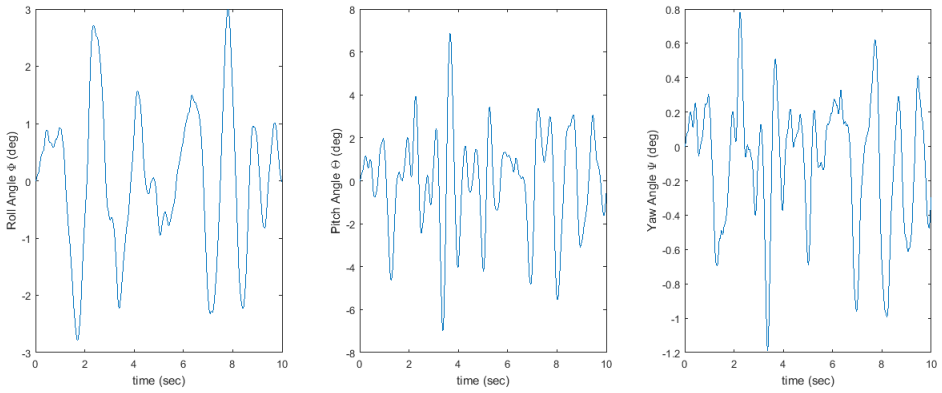

Figure 6.7: Nonlinear Hover Simulation: Euler Angles, ”Short Time”

However, for longer simulation times, Figures 6.9 and 6.10 show that the signals

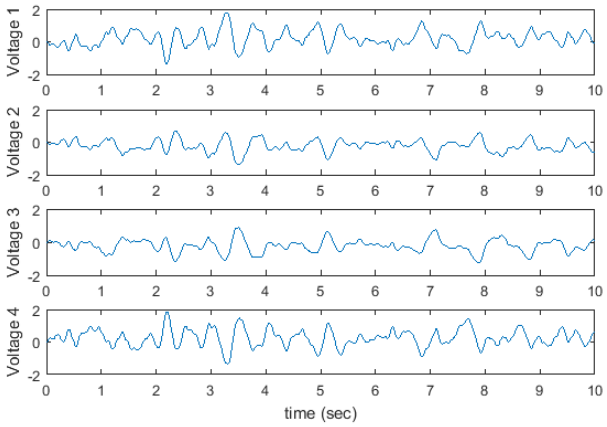

Figure 6.8: Nonlinear Hover Simulation: Control Voltages, ”Short Time”

diverge and reveal an instability.The cause was found when examining the poles of the synthesized controller. Two of the poles were found to have positive real parts, causing the controller to be unstable. It is believed the Bilinear Transformation is not at fault. Instead, an initial hypothesis is that the numerics of the linearization and synthesis have caused RHP zeros in the plant model, and these have induced unstable poles in the controller. More investigation into this problem is planned for future work.

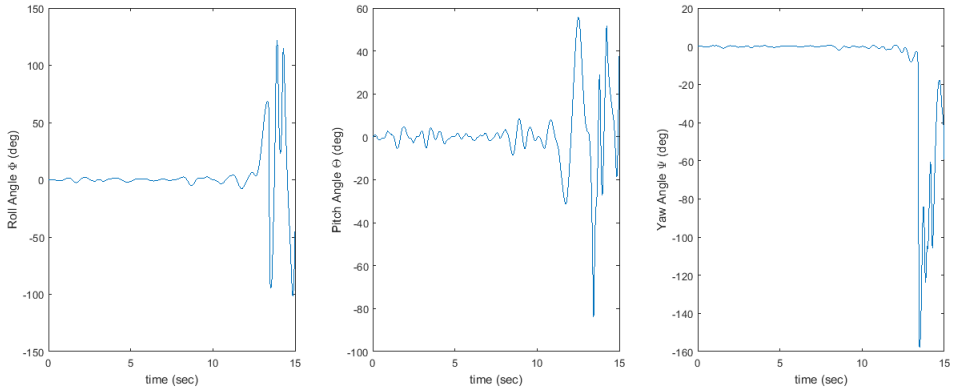

Figure 6.9: Nonlinear Hover Simulation: Euler Angles, ”Long Time”

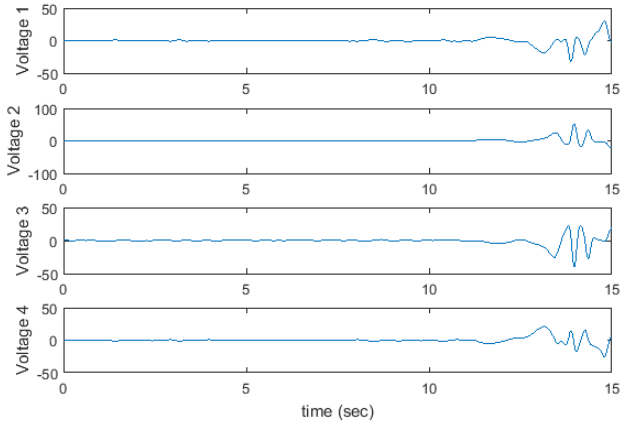

Figure 6.10: Nonlinear Hover Simulation: Control Voltages, ”Long Time”

7 CONCLUSION

In conclusion, design and modeling methods for sUAS which facilitate the synthesis of modern controllers were presented. The methods were chosen in order to be rapidly implementable, provide sufficient accuracy, and reduce cost. The H_∞ control design method was chosen because it allows rapid synthesis for MIMO systems without requiring gain tuning or full-state feedback.

Simulink was chosen as the modeling environment and SimMechanics was used to quickly model the aircraft dynamics. The aerodynamics of the vehicle were modeled using Athena Vortex Lattice. Custom scripts were created generate aerodynamic look-up tables which were applied to the nonlinear model. The Propeller aerodynamics were modeled by integrating the sectional lift and drag equations along the span of the blade. Once again, look-up tables were created for application in the nonlinear simulation. The BLDC motor was modeled by using parameter estimation to match the performance of a BDC motor model to experimental data taken from the BLDC motor.

H_∞ controllers were synthesized for cruise and hover configurations of a tiltrotor sUAS with the following results:

1. The cruise controller achieved acceptable performance by rejecting gust disturbances on the non-linear model.
2. The Hover controller was synthesized with the help of a bilinear transformation but is unstable. The solution to the instability will be the topic of future work.

REFERENCES

[1] B. Kada and Y. Ghazzawi, "Robust pid controller design for an uav flight control system," World Congress on Engineering and Computer Science, vol. 2, 2011.

[2] S. Franko, ""lqr based trajectory control of full envelope, autonomous helicopter," n Proceedings of the World Congress on Engineering, vol. 1, 2009.

[3] J. Lopez, R. Dormido, S. Dormido, and J. P. Gomez, "A robust h_∞ controller for an uav flight control system," The Scientific World Journal, 2015.

[4] U. University of Sydney, "Vortex lattice method (3-d)," 2016.

[5] J. Moran, An Introduction to Theoretical and Computational Aerodynamics. International series of monographs on physics, John Wiley & Sons, 1984.

[6] H. Youngren and M. Drela, "Athena vortex lattice method," 2016.

[7] J. D. Anderson, Fundamentals of Aerodynamics, Fifth Edition. McGraw-Hill, 2011.

[8] Messner and D. Bill Tilbury, "Dc motor speed: Simulink modeling," 2016.

[9] A. Turevskiy and MathWorks, "Estimating dc motor parameters," 2016.

[10] G. E. Dullerud and F. G. Paganini, A Course in Robust Control Theory: A Convex Approach. Springer, 2010.

[11] MathWorks, "Mathworks online documentation: linmod()," 2016.

[12] MathWorks, “Mathworks online documentation: Dryden wind turbulence model (continuous),” 2016.

[13] R. Y. Chiang and M. G. Safonov, “h_∞ synthesis using a bilinear pole shifting transform,” Journal of Guidance, Control, and Dynamics, vol. 15, no. 5, pp. 1111–1117, 1992.